QUINN EMANUEL URQUHART & SULLIVAN, LLP
Charles K. Verhoeven (Bar No. 170151)
charlesverhoeven@quinnemanuel.com
Melissa Baily (Bar No. 237649)
melissabaily@quinnemanuel.com
Lindsay Cooper (Bar No. 287125)
lindsaycooper@quinnemanuel.com
50 California Street, 22nd Floor
San Francisco, California 94111-4788
Telephone: (415) 875-6600
Facsimile: (415) 875-6700

Attorneys for GOOGLE, LLC

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

SONOS, INC., Plaintiff, vs. GOOGLE LLC, Defendant.	CASE NO. 3:20-cv-06754-WHA Related to CASE NO. 3:21-cv-07559-WHA **GOOGLE'S MOTION FOR SUMMARY JUDGMENT PURSUANT TO THE COURT'S PATENT SHOWDOWN PROCEDURE** The Hon. William H. Alsup Date: June 9, 2022 Time: 8:00 a.m. Location: Courtroom 12, 19th Floor

TABLE OF CONTENTS

Page

TABLE OF AUTHORITIES

Page

CASES

STATUTES

NOTICE OF MOTION AND MOTION FOR SUMMARY JUDGMENT

TO ALL PARTIES AND THEIR ATTORNEYS OF RECORD:

PLEASE TAKE NOTICE THAT, pursuant to the Court's Patent Showdown Scheduling Order (Dkt. 68), on June 9, 2022, at 8:00 a.m., or as soon thereafter as the matter may be heard, in Courtroom 12, 19th Floor, of the San Francisco Courthouse, 450 Golden Gate Avenue, San Francisco, California 94102, before the Honorable William Alsup, Google LLC ("Google") will and hereby does move for an order granting summary judgment on the following grounds: (i) Google does not infringe claim 13 of U.S. Patent No. 9,967,615 (the "'615 Patent"); (ii) claim 13 of the '615 Patent is invalid under 35 U.S.C. §§ 102 and 103; and (iii) Google does not infringe claim 1 of U.S. Patent No. 10,848,885 ("'885 Patent").

U.S. Patent No. 9,967,615 Claim 13

13[pre]. A tangible, non-transitory computer readable storage medium including instructions for execution by a processor, the instructions, when executed, cause a control device to implement a method comprising:

13.1 causing a graphical interface to display a control interface including one or more transport controls to control playback by the control device;

13.2 after connecting to a local area network via a network interface, identifying playback devices connected to the local area network;

13.3 causing the graphical interface to display a selectable option for transferring playback from the control device;

13.4 detecting a set of inputs to transfer playback from the control device to a particular playback device, wherein the set of inputs comprises: (i) a selection of the selectable option for transferring playback from the control device and (ii) a selection of the particular playback device from the identified playback devices connected to the local area network:

13.5 after detecting the set of inputs to transfer playback from the control device to the particular playback device, causing playback to be transferred from the control device to the particular playback device, wherein transferring playback from the control device to the particular playback device comprises:

(a) causing one or more first cloud servers to add multimedia content to a local playback queue on the particular playback device, wherein adding the multimedia content to the local playback queue comprises the one or more first cloud servers adding, to the local playback queue, one or more resource locators corresponding to respective locations of the multimedia content at one or more second cloud servers of a streaming content service;

(b) causing playback at the control device to be stopped; and

(c) modifying the one or more transport controls of the control interface to control playback by the playback device; and

13.6 causing the particular playback device to play back the multimedia content, wherein the particular playback device playing back the multimedia content comprises the particular playback device retrieving the multimedia content from one or more second cloud servers of a streaming content service and playing back the retrieved multimedia content.

U.S. Patent No. 10,848,885 Claim 1

[1.pre] A first zone player comprising:

[1.1] a network interface that is configured to communicatively couple the first zone player to at least one data network;

[1.2] one or more processors;

[1.3] a non-transitory computer-readable medium; and

[1.4] program instructions stored on the non-transitory computer-readable medium that, when executed by the one or more processors, cause the first zone player to perform functions comprising:

[1.5] while operating in a standalone mode in which the first zone player is configured to play back media individually in a networked media playback system comprising the first zone player and at least two other zone players:

(i) receiving, from a network device over a data network, a first indication that the first zone player has been added to a first zone scene comprising a first predefined grouping of zone players including at least the first zone player and a second zone player that are to be configured for synchronous playback of media when the first zone scene is invoked; and

(ii) receiving, from the network device over the data network, a second indication that the first zone player has been added to a second zone scene comprising a second predefined grouping of zone players including at least the first zone player and a third zone player that are to be configured for synchronous playback of media when the second zone scene is invoked, wherein the second zone player is different than the third zone player;

[1.6] after receiving the first and second indications, continuing to operate in the standalone mode until a given one of the first and second zone scenes has been selected for invocation;

[1.7] after the given one of the first and second zone scenes has been selected for invocation, receiving, from the network device over the data network, an instruction to operate in accordance with a given one of the first and second zone scenes respectively comprising a given one of the first and second predefined groupings of zone players; and

[1.8] based on the instruction, transitioning from operating in the standalone mode to operating in accordance with the given one of the first and second predefined groupings of zone players such that the first zone player is configured to coordinate with at least one other zone player in the given one of the first and second predefined groupings of zone players over a data network in order to output media in synchrony with output of media by the at least one other zone player in the given one of the first and second predefined groupings of zone players.

Google respectfully moves for summary judgment on three issues:

No infringement of Claim 13 of the '615 patent: In 2013, Sonos reached out to Google for help integrating its speakers with Google Play Music. The parties subsequently entered into a collaboration agreement, and Google shared with Sonos a novel system for permitting users to store queues of their musical selections in the cloud. When Google revealed its "cloud queue" plans to Sonos's lead engineer, Tad Coburn, he expressed excitement and called the development "very interesting." Mr. Coburn found Google's cloud queue technology so interesting that he attempted to claim the invention for himself in multiple patents, including one of the non-showdown patents that Sonos asserts against Google in this case (the '033 patent in which Sonos amended the claims in 2019 to add a "remote playback queue" limitation). By doing so, Sonos and Mr. Coburn misappropriated Google's technology and violated the terms of the parties' collaboration agreement. With respect to the '615 patent at issue in the showdown, however, this history illustrates the frivolous nature of Sonos's claims against Google, because that patent covers a "***local*** playback queue on a particular playback device" rather than the remote ***cloud*** queue that Google invented.

Sonos knows full well that Google does not implement the claimed local playback queue, which requires storing the playback queue at a local playback device. Instead, Google's accused devices request songs for playback from a cloud queue one-by-one. Because there is no local playback queue, Sonos argues that caching an identifier for an individual cloud queue item on the playback device transforms a cloud queue into a local playback queue. But this argument eliminates the "***local*** playback queue" from the claims, which Sonos notably added to secure issuance after years of battling the patent examiner at the PTO.

Google does not infringe claim 13 for at least two other reasons. Namely, Google does not store the claimed "resource locators" in any alleged "local playback queue," nor does Google add any "multimedia content" to that queue. These provide additional, independent, bases to grant summary judgment of non-infringement for claim 13, and they are described in detail below.

Invalidity of claim 13 of the '615 patent based on Google prior art: As discussed above, by 2013 Google began to transition its applications to a "cloud queue." Prior to this transition, however, Google's prior art products used a conventional "local playback queue" architecture for

managing music playback. In fact, Google developed this system for its YouTube Remote product in 2010, prior to the '615 patent priority date, and therefore Google's YouTube Remote product renders claim 13 of the '615 patent invalid.

The YouTube Remote allowed a user to queue up music on his or her mobile device, then transfer playback of that queue to one or more televisions. Notably, the YouTube Remote prior art product is a direct ancestor of the YouTube product Sonos accuses of infringement, and therefore it aligns with Sonos's infringement theories in the same ways. The key difference is that where the accused YouTube applications use Version 3 of the Mobile Device eXperience (MDx) protocol that implements a cloud queue, the prior art YouTube Remote used Version 1 that implements a local queue. Thus, Google's current (remote queue) products cannot infringe the patent, but its prior art (local queue) products invalidate that same patent.

No infringement of claim 1 of the '885 patent: The '885 patent covers an alleged advancement on speaker grouping, which allows users to set up "zone scenes" that carry out a user's chosen "common theme" for the speakers in their homes. Google's products, however, merely use generic and conventional speaker groups and do not implement Sonos's claimed "zone scenes." The parties previously disputed the meaning of the "zone scene" term in the Western District of Texas, with Sonos arguing that a "zone scene" only needed to be a predefined grouping of speakers, and Google arguing that the patent coined the new term "zone scene" to specifically refer to setting a particular "common theme" for a user's speakers. The court agreed with Google's proposed construction, clearly setting Google's products apart from the patent because the accused speaker products do not use a "common theme." However, relying on the fact that this case was transferred from the Western District, and because Sonos received an unfavorable construction there, Sonos attempts to ignore the operative claim construction order. But, as Google explained in its claim construction briefing, that construction is correct, and Sonos is indeed bound by that order because it affirmatively chose not to challenge it before this Court.

There is no serious dispute that Google's products do not include the claimed "common theme." Nevertheless, Sonos argues that the "common theme" might be whatever a user has in their mind at the time that they create a generic speaker group. This position is legally fraught because

courts across the nation, including the Federal Circuit, have squarely held that importing a mental state into the infringement analysis or claim construction is prohibited. Sonos also argues that any time a user assigns a name to a speaker group, this might also reflect the "common theme" that they intend, but this argument fares no better because the patent describes naming speakers separately from a "common theme." Under the proper construction of "zone scene," there is no genuine dispute that Google's products do not infringe claim 1 of the '885 patent, and therefore summary judgment of noninfringement is warranted.

I. STATEMENT OF ISSUES TO BE DECIDED

1. Whether Google is entitled to summary judgment that Google does not infringe claim 13 of the '615 Patent with respect to the accused products;

2. Whether Google is entitled to summary judgment that claim 13 of the '615 Patent is anticipated by the YouTube Remote prior art system, or rendered obvious in further view of the general knowledge of a person of skill and/or the YouTube Remote patent (U.S. Patent No. 9,490,998); and

3. Whether Google is entitled to summary judgment that Google does not infringe claim 1 of the '885 Patent with respect to the accused products.

II. LEGAL STANDARD

A court may grant summary judgment where the movant shows "there is no genuine issue as to any material fact and that the movant is entitled to judgment as a matter of law." *Celotex Corp. v. Catrett*, 477 U.S. 317, 322 (1986). Upon showing "that there is an absence of evidence to support the nonmoving party's case," the moving party's burden is met. *Id.* at 326.

III. ARGUMENT

A. Google Does Not Infringe Claim 13 of the '615 Patent

Claim 13 is directed to a "control device" that controls playback of a "local playback queue on [a] particular playback device." Here, Sonos alleges that a device (*e.g.*, a smartphone) running a YouTube or Google Play Music ("GPM") application is a "control device" that can control playback of a "local playback queue" stored on an alleged "playback device" (*e.g.*, a TV or speaker).

Google's accused systems, however, have long since moved away from a "local playback

queue" in favor of a playback queue stored in the cloud (*i.e.*, a "cloud queue"). Indeed, Sonos's infringement allegations against GPM are directed to the very same cloud queue functionality the parties developed during a collaboration from 2013 to 2015. Ex. 1 (Bhattacharjee Decl.), ¶112. In November of 2013, Google told Sonos—including the named inventor on the '615 patent, Sonos engineer Tad Coburn—that Google was considering using "a more ***cloud queue*** centric model," to which Mr. Coburn responded that "[t]he idea of ***moving the playlist to the cloud*** is very interesting, but will definitely complicate things." Dkt. No. 123-3 (SAC), ¶25. In the months that followed, Mr. Coburn continued to express excitement regarding Google's cloud queue idea, including asking Google to share its cloud queue API design—which Google did. *Id.*, ¶¶26-30. Sonos now accuses the Cloud Queue API functionality in GPM and cloud queue functionality in YouTube, despite knowing that Google's cloud queue design is different than the claimed "local playback queue."[1]

1. The Accused YouTube Applications Do Not Infringe

Claim 13 requires a "control device" to transfer playback to a "particular playback device." The step of transferring playback includes three limitations (annotated with numerals below):

> **[1]** causing one or more first cloud servers to add multimedia content **[2]** to a local playback queue on the particular playback device, wherein adding the multimedia content to the local playback queue comprises the **[3]** one or more first cloud servers adding, to the local playback queue, one or more resource locators corresponding to respective locations of the multimedia content at one or more second cloud servers of a streaming content service;

'615 patent, Claim 13. Thus, to demonstrate infringement Sonos must show that the system has "a local playback queue" and that the system adds both "multimedia content" and "resource locators" to that local playback queue. The accused YouTube systems do not meet any of these limitations.

2. The Accused YouTube Applications Use A Cloud Queue, Not A "Local Playback Queue On The Particular Playback Device"

The accused YouTube system does not use a local playback queue on the particular playback

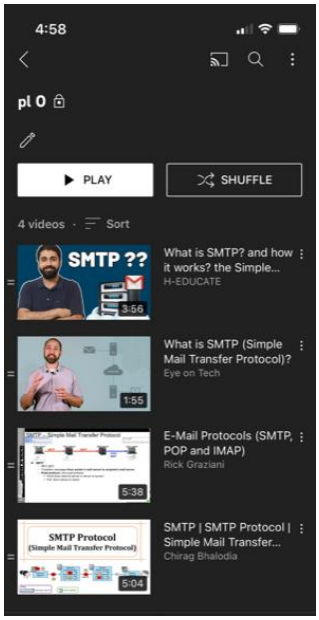
[1] Although each of Sonos's infringement theories fail because the accused applications do not have a "local playback queue," the two sets of accused products, the YouTube applications and Google Play Music, operate differently and are therefore addressed separately below. Sonos has also manufactured numerous scattershot literal and doctrine of equivalents infringement theories for each product, which Google addresses in turn.

device and therefore cannot infringe claim 13. Instead, the accused YouTube system uses a "cloud queue" in which the "playback queue" is stored remotely on an "MDx" server. A playback device plays from the cloud queue by requesting cloud queue items one-by-one, not by maintaining the claimed local queue. Ex. 1 (Bhattacharjee Decl.), ¶¶ 63-66.

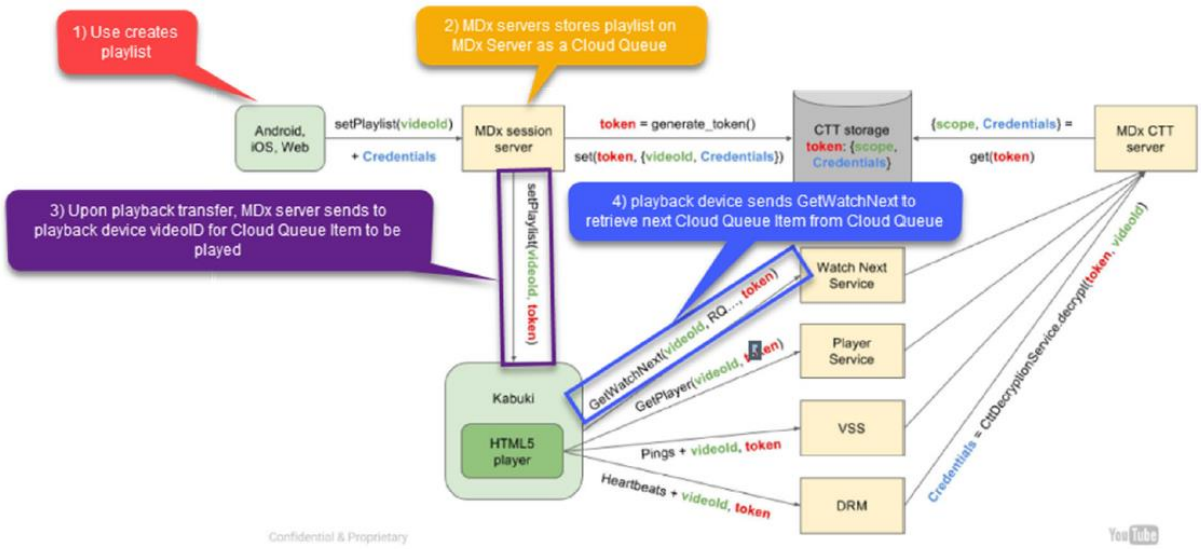
Sonos accuses Google's MDx protocol as providing the local playback queue. The MDx protocol does manage playback on a device in YouTube; however, only older versions of that protocol stored the playback queue on the playback device. When Google transitioned to MDx Version 3, it eliminated the playback queue on the playback device in favor of a cloud queue in the maintaining it on the MDx server. *See* Ex. 2 at GOOG-SONOSWDTX-00041748 ("the queue is now maintained on the ***MDx server and not the TV***."); Ex. 3 at GOOG-SONOSWDTX-00039988 ("MDx is the first server-backed ***Cloud queue*** at YT"). Indeed, Google's documents repeatedly explain that when a mobile device "Casts" (*i.e.*, sends) playback to a playback device, the queue is stored in a "remote queue"—***not a local queue***. Ex. 4 at GOOG-SONOSWDTX-00039798 ("[w]hen Casting, the ***queue is persisted as a server-side 'remote queue'***"), GOOG-SONOSWDTX-00039799 ("Casting use case ***stores the queue in YouTube servers as a "Remote Queue" playlist***."), GOOG-SONOSWDTX-00039800 ("MDx Session Server manages the '***Remote Queue***' playlist," and users "make queue edit operations (add to queue, re-order, remove from queue, etc.) through the MDx Session Server.").[2]

[2] Version 3 of the MDx protocol was created by January 2014. Ex. 5 (MDx Communication Protocol v3) at GOOG-SONOSWDTX-00037243. Thus, the decision to move to a Cloud Queue in MDx came just months after Google informed Sonos that it was moving to a Cloud Queue in GPM. See Section III.A.1.

Using the accused YouTube applications, a user may select and queue up music or other media items and save them to a playlist (shown on the right) that is sent to an MDx server in the cloud and stored as a "cloud queue." Ex. 1 (Bhattacharjee Decl.), ¶¶51-53. The remote "cloud queue" is the only "queue" Sonos has identified in the accused YouTube system, and indeed it does have qualities of a playback queue. For example, this cloud queue is an ordered list of multimedia items (referred to as "cloud queue items") selected by the user for playback, which tracks both Google's construction and the plain meaning of the term "playback queue." Dkt. No. 200 (Google Responsive CC Br.) at 11; *Id.*, ¶68. Consistent with the '615 patent's description of a queue, the cloud queue may be edited and managed by the user (*e.g.*, users may add or remove cloud queue items, reorder those items, or clear the cloud queue altogether). *Id.*, ¶¶65, 73.



Critically, the playlist shown above is not stored on a playback device. Instead, with reference to the annotated image below, a user [1] creates a playlist; [2] the playlist is stored on the MDx server as a Cloud Queue; [3] when a user tells the playback device to play back a video from a playlist, the MDx server sends the playback device an identifier, called a videoId, for the cloud queue item that should be played; and [4] each time a playback device wants the next cloud queue item it sends a "GetWatchNext" request to retrieve the next videoId from the Cloud Queue.



Id., ¶60 (annotating Ex. 6 at GOOG-SONOSWDTX-00039491).

Further, the accused YouTube system only retrieves cloud queue items for playback one-by-one, further illustrating that there is no "local playback queue." Ex. 1 (Bhattacharjee Decl.), ¶¶52-60. If the playback device stored the "playback queue" locally, it could play it back without needing to retrieve each cloud queue item individually because they would already be available. *Id.*.

Because Google's products do not use a "local playback queue on the particular playback device,"[3] Sonos chose to "throw everything at the wall to see what sticks"—devising three alternative theories for what might possibly satisfy the "local playback queue" limitation. Each of these theories is flawed because they accuse Google's cloud queue, not "a local playback queue on the particular playback device."

Sonos's First and Second Theories: Sonos's first theory accuses "each of Google's data variables currentVideoIdDeprecated and currentWatchEndPoint.videoID" of being a "local playback queue," while Sonos's second theory accuses these variables "in combination with Google's data variable upNextVideoID." *Id.* at 40. Both of these theories fail because they misunderstand how Google's products work.

When the MDx server sends a playback device the videoId for a cloud queue item that should be played, the playback device stores that videoId in the variables "currentVideoIdDeprecated" or "currentWatchEndPoint.videoID," both of which are not queues themselves, but rather each is a variable containing a single videoId for the current cloud queue item.[4] Ex. 1 (Bhattacharjee Decl.), ¶¶69-71. After the playlist stored in the cloud has been exhausted, the YouTube servers may sometimes send the playback device another videoId for a recommended "autoplay" video. *Id.*, ¶71.

---

[3] Tellingly, after Google informed Sonos that its accused instrumentalities do not use a "local playback queue on the particular playback device" at the outset of this case (Dkt. No. 41, ¶ 25), Sonos initially refused to identify the claimed "local playback queue" in the accused device. (*See* Dkt. 86-4 [Google Ltr. Brief to Compel]). It was only after the Court granted Google's motion to compel that Sonos disclosed what it was accusing as the "local playback queue." *See* Dkt. 99.

[4] The variable currentWatchEndPoint.videoID is a replacement for currentVideoIdDeprecated–hence the name. The source code, for instance, includes a comment which states that currentVideoIdDeprecated should not be used for new code: "Note: DO NOT USE this for new code" (remote.ts at 117-123). And elsewhere the comments explain that currentVideoIdDeprecated is used as a "fallback" alternative if currentWatchEndpoint.videoId does not exist (remote.ts at 1369-76). Ex. 1 (Bhattacharjee Decl.), ¶71.

The videoId for this autoplay video is stored in the accused upNextVideoID variable. *Id.*, ¶71.[5]

Neither theory satisfies the local playback queue limitation under either Google's proposed construction or the plain meaning of the term. The variables currentVideoIdDeprecated and currentWatchEndPoint.videoID are not queues; they merely store a single, static item (the current videoId)—not an "ordered list of multimedia items selected by the user for playback." *Id.*, ¶71. The playlist where the accused products store the ordered list of items to be played is in the cloud queue on the MDx server rather than any "local" playback queue. *Id.*, ¶¶72-74. Moreover, the addition of the upNextVideoID in Sonos's second theory—which also stores a single, static item—does not alter this fact because the ***list*** of recommended videos, like the playlist, is still stored in the cloud on the YouTube servers. *Id.*, ¶¶72-74. In other words, Sonos has not provided evidence that any alleged playback device stores the playback ***queue*** because none of the variables Sonos accuses are—by themselves or together—an ordered list of multimedia items selected by the user for playback. Rather, Sonos merely argues that Google locally caches a single cloud queue item that is currently playing and, after the playlist is exhausted, an identifier for a single recommended video/song.

Sonos's first and second theories depend on the faulty premise that a variable containing a ***single*** cloud queue item is a "playback queue." This is an unreasonable interpretation of the term "playback queue," to a person of skill in the art, Ex. 1 (Bhattacharjee Decl.), ¶¶71-73, and it is inconsistent with the '615 patent, Dkt. No. 200 at 11-18. The '615 patent teaches that a "queue" is an ordered list of tracks ('615 patent at 16:35-40), and that it may be "edit[ed]/[manag[ed]" by, for instance, allowing a user to "add, delete, and so on from the queue" ('615 patent at 16:52-62). Relatedly, the '615 patent refers to the "Sonos Controller" as an exemplary "control device" ('615 patent at 5:2-11), and the Sonos Controller manuals similarly define a "queue" as the "list of tracks" selected by the user (Dkt. No. 200-12 at 4-2), provide that the queue can be populated with a variable number of items and edited and managed by adding, removing, and reordering tracks (*Id.* at 4-10 to 4-12). The individual variables that Sonos cherry-picks in Google's system are not capable of any

[5] Notably, the accused upNextVideoID variable exists in the system for only a few microseconds. *Id.*, ¶77-78.

of these queuing functions.

Sonos argued during claim construction that a "playback queue" is a structure that can be populated with "zero, one or multiple media items at any given time." Dkt. No. 184 (CC Br.) at 12. In contrast, the data variables that Sonos accuses can each only hold one item (the current videoId or the recommended videoId) and collectively only two items—they cannot be populated with zero, one or many media items at a given time. Ex. 1 (Bhattacharjee Decl.), ¶¶71-73. For this reason, referring to these variables as a "local playback queue" is inconsistent with the plain meaning. *Id.*, ¶72. A "playback queue" can also be managed and edited by the user—*e.g.*, items can be added to the queue to make it larger, they can be removed from the queue to make it smaller, or they can be re-ordered. *Id.*, ¶73. The local variables that Sonos accuses cannot, and they at most cache individual items ***from*** a playback queue—they are not the playback queue itself. *Id.*, ¶74.

Sonos's argument that the upNextVideoID is a queue fails for additional reasons, such as the fact that the accused upNextVideoID variable exists for only a few milliseconds, whereas a person of skill in the art would understand that a queue must persist for long enough to actually function as a queue by arranging multimedia items for playback. *Id.*, ¶¶76-77. Further, the videoID for an autoplay video that is stored in upNextVideoID is neither part of the "local playback queue," nor is it "selected by the user." *Id.*, ¶78. Rather, an autoplay video is a recommended video that is selected by the YouTube servers after the cloud queue has been exhausted. *Id.*, ¶¶78-79. Moreover, the claim language makes clear that the "local playback queue" must exist as part of the step of transferring playback: "transferring playback from the control device to the particular playback device comprises: causing one or more first cloud servers to add multimedia content to a local playback queue." '615 Pat. Cl. 13. The upNextVideoID variable is added to the playback device after the device has ***exhausted*** the playlist, and thus well after playback has been transferred.

Sonos's Third Theory: Sonos's third theory accuses "WatchNextResponse" of being a "local playback queue." Far from being a "local playback queue," WatchNext Response is primarily used to obtain information needed to populate the user interface (including metadata, comments, advertisements, information about the number of likes or dislikes for the current video, text describing the number of videos on the channel, etc.), and includes among this information the

videoId for the next cloud queue item that should be played. Ex. 1 (Bhattacharjee Decl.), ¶¶80-84. A response that contains such a hodge-podge of information is not a "playback queue," and Sonos has not identified anything within the response that can be considered a "playback queue." *Id.*, ¶82-83. Indeed, a person of skill in the art would understand that a playback queue is stored in a data structure by linking together different multimedia items (*e.g.* URLs corresponding to the location of the multimedia) in a particular order using linked lists, arrays, vectors, or other well-known data structures. *Id.*, ¶82. Sonos has not pointed to any such list structure. *Id.*, ¶83. Further, a playback queue persists beyond the playback of the current media, unlike the WatchNextResponse which is received anew each time a new song or video is retrieved from the cloud queue. *Id.*, ¶84. And while a playback queue can be edited and managed, with items added, removed, or reordered in the queue, the WatchNextResponse does not permit this type of queue manipulation because it contains a static set of items. *Id.*, ¶84.

Sonos's Doctrine of Equivalents ("DoE") Argument: In "exceptional cases," a patent owner that cannot show literal infringement may argue instead that the system infringes under DoE. *See Amgen Inc. v. Sandoz Inc.*, 923 F.3d 1023, 1029 (Fed. Cir. 2019) ("The doctrine of equivalents applies only in exceptional cases and is not simply the second prong of every infringement charge, regularly available to extend protection beyond the scope of the claims.") (cleaned up). This is not such an "exceptional" case, and Sonos may not rely on DoE here for at least two reasons.

First, Sonos added the "local playback queue" claim element in a narrowing amendment during prosecution and therefore surrendered any right to equivalents of that claim element. During a prolonged battle with the patent examiner, including three rejections, Sonos finally amended the claims of the '615 patent to include the "local playback queue" term (along with other narrowing amendments), and the examiner later issued the patent. Ex. 7 (2016-10-25 Claim Amendment) at 1. Supreme Court law is clear that such a narrowing amendment results in a surrender of any equivalents of this term: "When the patentee has chosen to narrow a claim, courts may presume the amended text was composed with awareness of this rule and that the territory surrendered is not an equivalent of the territory claimed." *Festo Corp. v. Shoketsu Kinzoku Kogyo Kabushiki Co.*, 535 U.S. 722, 741 (2002). Accordingly, the *Festo* presumption applies, and Sonos is not entitled to

equivalents on this claim element.

Second, Sonos cannot show that its claimed equivalent has only "insubstantial differences" compared to a literal reading of the claim language. *Bio-Rad Lab'ys, Inc. v. 10X Genomics Inc.*, 967 F.3d 1353, 1367 (Fed. Cir. 2020) (holding that when "two alternatives exist that are very different from each other [they] cannot be equivalents for infringement purposes."). Google's approach—using a cloud queue—is fundamentally different than the "local playback queue" system that Sonos claimed. *See* Ex. 1 (Bhattacharjee Decl.), ¶¶86-88. Of course, to implement a "cloud queue" there will likely be data cached at the device for speed and efficiency, but this does not transform every non-infringing "cloud queue" implementation into a "local playback queue" implementation. A cloud queue provides centralized storage of the playback queue, and as a result it can support playback and interactions from a large number of users and can be shared and synchronized with many devices. The cloud playback queue is not restrained by the capabilities of the playback device's hardware (*e.g.*, memory), and the queue is not lost when a local playback device fails. Therefore a local and remote playback queue are "very different" ways of using queues in the multimedia context and are not equivalents. *Id.*, ¶¶86-89.

3. The Accused YouTube Applications Do Not Store "Multimedia Content" In Any Alleged Playback Queue

The claims require an accused system to "add multimedia content to a local playback queue." In addition to the fact that the accused YouTube Applications do not include a "local playback queue" (*supra* III.A.2), Sonos cannot show the accused YouTube system adds "multimedia content" to any alleged "playback queue."

Sonos appears to argue that "multimedia content" may be a "videoID" that serves as an identifier for the media. But an ***identifier*** of multimedia is not the multimedia content itself. Rather, "content" is the actual multimedia file or information and data within that file that may be played by the playback device—for example a song or video. Ex. 1 (Bhattacharjee Decl.), ¶¶90-91. A videoID, by contrast, is merely an identifier for the content that is typically only a short string of characters; for example, a videoId could be "n_yx_BrdRF8." Ex. 5 [MDx Communication Protocol v3] at 4.

Indeed, the claims and specification confirm that an identifier, such as a videoId, is not "multimedia content." Claim 20 recites "causing an identifier of the multimedia content to be added to the playback queue," thereby distinguishing an identifier of the multimedia content from the multimedia content itself. Similarly, Claim 13 recites "playing back the retrieved multimedia content." A POSITA would understand that a videoId cannot be played back because it is merely a short string of characters that identifies the video and is not the audio or video content itself. Ex. 1 (Bhattacharjee Decl.), ¶¶92-94. The specification of the '615 patent also refers to the "multimedia content" as the actual music or video, and distinguishes it from an identifier which may also be stored in the queue. '615 patent at 12:58-67 ("Songs and/or other multimedia can be retrieved from the Internet"), 13:54-57 ("play music, audio, video and/or other multimedia content."). Accordingly, Sonos has not identified any "multimedia content" that is added to a "local playback queue" because multimedia identifiers are not multimedia content under the plain language of the claims and the description in the specification.

4. The Accused YouTube Applications Do Not Store "Resource Locators" In Any Alleged Playback Queue

The asserted claim further requires "one or more resource locators" that are "add[ed] to the local playback queue." In its infringement contentions, Sonos provides two theories for this limitation: (1) that the "resource locators" are a "videoID," and (2) that the resource locators are a URL stored within a "dashManifest." The first theory fails because the identified "videoID" is not a resource locator and the second theory fails because no URL in the dashManifest is added to a local playback queue.

The accused videoID in Sonos's first theory cannot be the claimed "resource locator" as it says nothing about where the media content is located. Using only the videoID, one has no way of knowing where the content is located. Ex. 1 (Bhattacharjee Decl.), ¶¶95-96. In fact, in the accused systems after a videoID is received, it must be mapped to a server from which to request the content. *Id.*, ¶¶96-97. The same videoID may be mapped to different servers depending on various conditions and circumstances. *Id.*, ¶¶96-97. As such, the multimedia content is stored in multiple locations, and those locations are not even known at the time the videoID is received. *Id.*, ¶97.

Recognizing that a videoID doesn't provide location information itself, Sonos instead argues that the "resource locator" need only "facilitate locating a resource" and can do so "indirectly." Dkt. 202 at 12 ("the 'resource locator' must still ***facilitate*** locating a resource (even if indirectly) and not merely identifying a resource.") (emphasis in original). Sonos's true construction (hidden behind the alleged "plain meaning") is therefore that a resource locator must "indirectly facilitate locating a resource." *See id.* Because Sonos has taken such an implausible view of the "plain meaning" of this term, Google proposed that "resource locator" be construed as an "address of a resource on the Internet" to crystalize the dispute, and Google showed that this construction is supported by and consistent with the intrinsic evidence in its claim construction briefing. Dkt. No. 200 at 18-21. Under Google's construction or the actual plain meaning of "resource locator," a videoID cannot meet this claim element. A videoId is simply a string of characters (*e.g.*, "n_yx_BrdRF8") that serves as an identifier for a media item. It is not an address and it does not "locate" the resource as required by the claims. Ex. 1 (Bhattacharjee Decl.), ¶¶95-98.

Sonos's second theory changes gears and accuses a URL stored in a "DashManifest." While this other URL that Sonos accuses is a "resource locator," it cannot satisfy the further requirement of the claim that the resource locator be "add[ed] to the local playback queue." As explained in Section III.A.2, Sonos has alleged that the "local playback queue is one or more of the data variables currentVideoIdDeprecated, currentWatchEndPoint.videoID, and upNextVideoID or the WatchNextResponse. The dashManifest is not added to or part of any of these alleged local playback queues. Ex. 1 (Bhattacharjee Decl.), ¶98. Instead, the dashManifest is a completely separate object that is used to define various parameters for video streaming. *Id.*, ¶98.

5. The Accused Google Play Music Application Does Not Infringe

Like the accused YouTube applications, Google Play Music (GPM) does not satisfy the "local playback queue" or "multimedia content" limitations either.

6. GPM Uses A "Cloud Queue," Not A "Local Playback Queue On The Particular Playback Device"

Playback of a queue in GPM is managed by Google's "Cloud Queue" API, which allows a user to construct a list of tracks and store them in a Cloud Queue for playback. Ex. 1(Bhattacharjee

Decl.), ¶¶99-112. Cloud queue is a service owned by Play Music's infrastructure team. First-party thick clients (Android, iOS, web) construct lists of tracks (queues) and store them in Cloud Queue, and then tell receivers (Chromecast, Sonos) what item to play. *Id.* Notably, Sonos now accuses of infringement the same Cloud Queue API functionality that Google developed with Sonos during their collaboration. Ex. 1 (Bhattacharjee Decl.), ¶112. At that time, rather than claiming the Cloud Queue API was a "local playback queue" as Sonos does now, the inventor of the '615 patent recognized that the Cloud Queue API was "moving the playlist to the cloud." Dkt. No. 123-5 at 2; *see also* Dkt. No. 123-7 (Sonos emailing Google about the status of the "queues in the cloud" project).

A general overview of the Cloud Queue API is shown below and described here. [redacted]

[redacted]

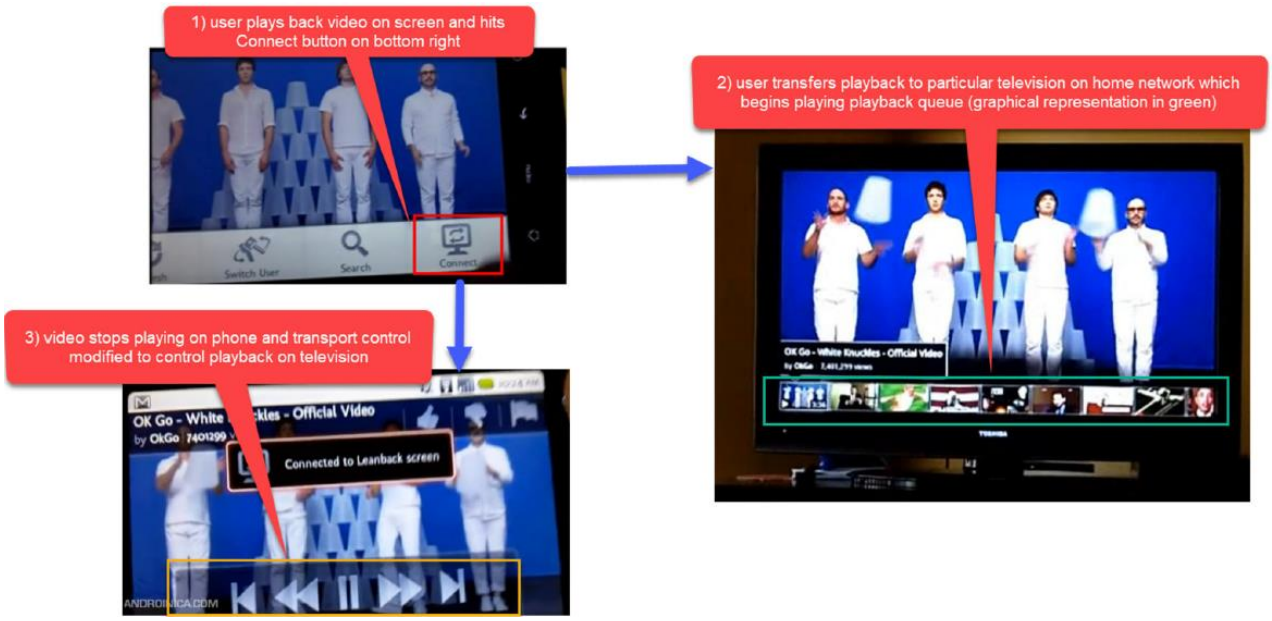
Sonos accuses the "ItemWindowResponse" of being a "local playback queue." The ItemWindowResponse, however, is merely a window of three locally cached items from the "playback queue"—it is not the "playback queue" itself. The "playback queue" is the complete "ordered list of multimedia items selected by the user for playback," which remains in the Cloud Queue. Dkt. No. 200 (CC Br.) at 11. Indeed, the cloud queue stored in the Cloud Queue servers contains a list of ItemIds corresponding to all the media items selected by the user and supports the features that would be associated with a playback queue, such as the normal playback order of the items, whether shuffle mode is enabled, the shuffled playback order of the items, and the playback modes (*e.g.*, repeat track). *Id.*, ¶104. The ItemWindowResponse, in contrast, contains a static set of items that do not support the same type of queue management. *Id.*, ¶¶107-111. The static set of items in the ItemWindowResponse also cannot be the "playback queue" because the itemWindowResponse contains only three items at a given time. *Id.*. The size of a playback queue in GPM may include tens of media items. *Id.*. These GPM playback queues are stored in the cloud queue; they cannot exist in the ItemWindowResponse because it is merely a windowed view of the previous, current and next media items in the playback queue which is stored in the cloud. *Id.*, ¶12.

7. GPM Does Not Store "Multimedia Content" In An Alleged Local Playback Queue

The claims require that the system add "multimedia content" to the "local playback queue." Sonos argues that the itemWindow response in GPM is the claimed "local playback queue." However, the only thing that Sonos identifies as being added to the itemWindow response is a URL. As discussed *supra*, a URL cannot be "multimedia content" because the claim requires that the system "play back the multimedia content" (*see* Claims Appendix, Claim 13.6) and there is no way to "play back" a URL.

B. An Earlier Version of the YouTube Application Invalidates Claim 13

Google released the YouTube Remote ("YTR") application on November 9, 2010, over a year before the '615 Patent's priority date. Ex. 9 (Bobohalma Decl.), ¶3.[6] Users of the YTR prior art could queue up YouTube videos on their phone by adding them to a playback "queue," as can be seen in the image of the YTR prior art on the right. Ex. 1 (Bhattacharjee Decl.), ¶129. A user could begin playback of the queue on the YTR application ("the control device") and then transfer playback to a particular television ("playback device") on the user's home Wi-Fi network by pressing a "Connect" button ("selectable option for transferring playback") on the graphical interface of the application. Thereafter, a user could use "transport controls" (*e.g.*, play, pause, skip) to control playback on the television. This process is shown in the images below from a video review of the YTR application uploaded to YouTube on November 14, 2010:



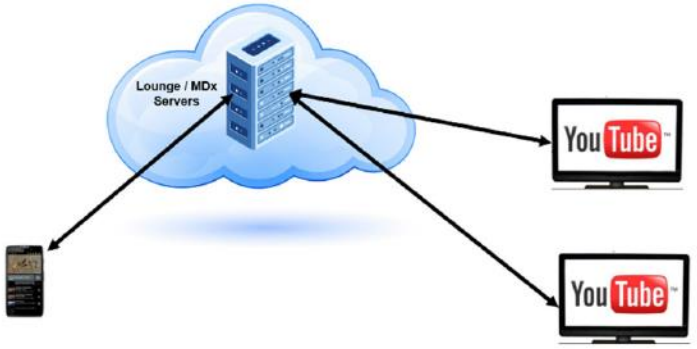
https://www.youtube.com/watch?v=EGdsOslqG2s ("YTR Video"); Ex. 1 (Bhattacharjee Decl.), ¶130, 136.

In fact, the protocol that manages transfer and control of playback from a prior art YTR

[6] YTR is therefore prior art under 35 U.S.C. § 102(a), (b), and (g).

application to a playback device is the MDx protocol—the same protocol Sonos accuses of infringement. Ex. 5. But whereas the accused applications use MDx Version 3, the YTR prior art used the older MDx Version 1. Ex. 9 (Bobohalma Decl.), ¶3. This distinction is significant because, as mentioned above, one of the changes that Google made in MDx Version 3 was to eliminate the playback queue on the playback device in favor of maintaining it in a Cloud Queue. Ex. 2 at GOOG-SONOSWDTX-00041748; Ex. 1 (Bhattacharjee Decl.), ¶¶49, 64 (showing YTR prior art stores queue on playback device).

Many aspects of the MDx architecture have remained the same over the years (shown on the right), with both the YouTube application and playback devices talking to one another through an MDx server. *Id.*, ¶¶128-145; Ex. 2. The table below compares the accused and prior art functionality for the limitations in Claim 13 relating to "transferring playback" and "playing" the local playback queue. *See* Claims Appendix, Limitations 13.5-13.6. To the extent Sonos's infringement contentions are credited, Sonos cannot dispute the YTR prior art discloses the same limitations. *01 Communique Lab'y, Inc. v. Citrix Sys., Inc.*, 889 F.3d 735, 742 (Fed. Cir. 2018) ("if a claim term must be broadly interpreted to read on an accused device, then this same broad construction will read on the prior art.").



Allegations In Infringement Contentions	YTR Prior Art
[1] transferring playback causes "MDx servers to send to a particular cast-enabled media player a 'set Playlist' message." At 37.	[1] transferring playback causes MDx servers to send playback device (television) a "setPlaylist" message.
Sonos's contentions incorrectly state the "setPlaylist" message includes "videoIDs" [plural], when in fact, the message includes only a single videoID for the video that should be played. Ex. 1 (Bhattacharjee Decl.), ¶179.	The setPlaylist message contains a comma separated list of videoID values representing the current playlist. Ex. 10 (YT Remote API) at 3; Ex. 1 (Bhattacharjee Decl.), ¶179.
[2] YouTube application "stop[s] its own playback of the multimedia content." At 32.	[2] YTR application stops its own playback of the multimedia content. Ex. 1 (Bhattacharjee Decl.), ¶139.
[3] YouTube application "modif[ies] one or more transport controls of its control interface	

such that the one or more transport controls function to control playback by the at least one particular Cast-enabled media player rather than playback by the Cast-enabled control device." At 32.	[3] YTR application modifies the transport controls (e.g., play, pause, skip, etc.) so that they function to control playback on the playback device (television). Ex. 1 (Bhattacharjee Decl.), ¶177-181.
[4] "The particular Cast-enabled media player retrieving the multimedia content from one or more cloud servers that differ from the one or more MDx servers (e.g., one or more CDN or "Bandaid" servers)." At 88.	[4] The playback device (television) retrieves YouTube videos from the Bandaid servers in Google's CDN. Ex. 1 (Bhattacharjee Decl.), ¶184; Ex. 9 Bobohalma Decl., 3; Ex. 11 Levai Decl., ¶¶3-4

Sonos disputes the YTR's disclosure of only two other limitations: Limitations 13.2 and 13.4. *See* Claims Appendix. The YTR prior art satisfies these limitations, or they are obvious.

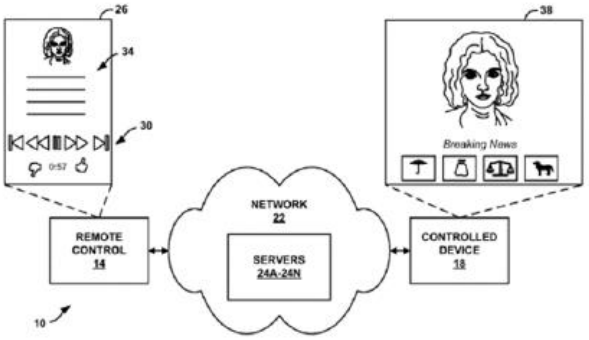
Anticipation: The YTR prior art discloses "identifying, via the control device, playback devices connected to the local area network" (Limitation 13.2). For example, an Android phone running the YTR application identifies playback devices (televisions) connected to a local area network. Ex. 1 (Bhattacharjee Decl.), ¶157-158. Indeed, in order for a YTR application to be paired to one or more televisions, the application and televisions need to be paired together over the Internet. Ex. 1 (Bhattacharjee Decl.), ¶158. This was accomplished by having each television connect to the Wi-Fi network, navigating to the website www.youtube.com/leanback, and logging each television into the same YouTube account that the YTR application is logged into. *Id.*, ¶158; GOOG-SONOSWDTX-00041837 ("The user opens Leanback. Leanback registers with the server…"). Each time a television was paired, it registered with the MDx server and the server sent a "loungeScreenConnected" message to the YTR application indicating that the television was connected and available to transfer playback. *Id.*, ¶158; GOOG-SONOSWDTX-00041837 (after television registers with the server, "[t]he server sends a loungeScreenConnected message to the remote."). Thus, the prior art YTR application identifies one or more televisions connected to the LAN by receiving loungeScreenConnected messages. *Id.*, ¶158.

The YTR prior art also discloses "detecting, via the control device, a set of inputs to transfer playback from the control device to a particular playback device, wherein the set of inputs comprises: (i) a selection of the selectable option for transferring playback from the control device and (ii) a selection of the particular playback device from the identified playback devices connected

to the local area network" (Limitation 13.4). To transfer playback using the YTR application the user may press the menu button (shown in green in the image on the right) which brings up a "Connect" button (shown in red) that the user may further select to transfer playback to a particular playback device. *Id.*, ¶159. The user's selection of menu and select are a "set of inputs" to transfer playback that are detected by the Android phone. *Id.*, ¶159-162. Additionally, the Connect button satisfies the two "selection" elements because it is both a "selectable option" for transferring playback and a selection of the particular playback device. *See Powell v. Home Depot U.S.A., Inc.*, 663 F.3d 1221, 1231-32 (Fed. Cir. 2011) (holding that the limitation "said cutting box interior in fluid communication with dust collection structure for collecting sawdust" could be literally met by a single "dust collection structure"). Thus, the YTR remote anticipates Claim 13.

Obviousness: To the extent Sonos contends that the Connect button is only a "selectable option for transferring playback" and that "a selection of the particular playback device from the identified playback devices" requires the ability to further select some (but not all) of the multiple televisions that may have been paired, it would have been at least obvious to modify the YTR remote so that upon pressing the Connect button the televisions that are paired with the YTR application are displayed and selected individually for transfer. Ex. 1 (Bhattacharjee Decl.), ¶163-173.

For example, Google filed a patent based on its work on the YTR prior art . Ex. 9 (Bobohalma Decl.), ¶4. Specifically, U.S. Patent No. 9,490,998 ("the YTR Patent") was filed on March 7, 2011 and claims priority to an earlier provision application filed on November 8, 2010.[7] The inventors of the YTR Patent are also inventors of the YTR prior art. Ex. 9 (Bobohalma



[7] The YTR Patent was filed before Sonos's alleged invention date (July 15, 2011) and is prior art.

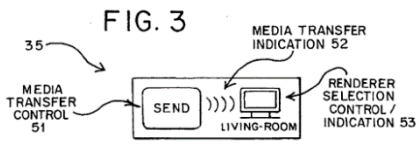
Decl.), ¶4. Figure 1 of the YTR Patent is shown above and illustrates a "Remote Control 14" (*e.g.*, a smartphone running the YTR application), connected to a "Controlled Device 18" (*e.g.*, a paired television) via "servers 24A-24N" (*e.g.*, the MDx servers). The YTR Patent expressly discloses that the "user interface" of the Remote Control may display the "previously paired controlled devices" so that a user may select and control "one or more paired controlled devices." YTR Patent at 10:62-11:6; *see also id*. at 8:11-12 ("Remote controls 62 and controlled devices 64 may be paired using a variety of techniques"). There can be no genuine dispute that it would have been obvious to combine the teachings of the YTR Patent with the YTR system. Both are directed to the same field, the same product, and involve the same inventors. Ex. 1 (Bhattacharjee Decl.), ¶167-169. In fact, Google implemented this feature in the YTR system prior to the effective filing date of the '615 patent (December 30, 2011), and released a version of the YouTube remote with the feature by January 25, 2012, as shown in the image on the right. *Id*., 169, Ex. 14.

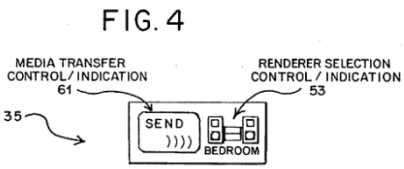
As another example, this limitation was also obvious in view of Google's Project Tungsten, Apple's AirPlay, and/or the Al-Shayk patent publication. Ex. 1 (Bhattacharjee Decl.), ¶172-173.

Project Tungsten: Google's Project Tungsten was unveiled at Google's annual developer conference on May 10, 2011. As demonstrated at the conference, with Project Tungsten a tablet was used to "direct music to one or more Tungsten boxes like the ones we have here [*i.e.*, the "Stage Left" or "Stage Right" output speakers shown in the image above]." https://www.youtube.com/watch?v=3SNPFPKS4U4 (video of developer conference, updated on May 10, 2011) at 3:40-4:00.

Apple Airplay: Similarly, in November of 2010 Apple released Airplay, which enabled a control device such as an iPhone or iPad to identify devices on the Wi-Fi network, and then allowed users to click on an "Airplay" icon to display an option to transfer playback to a particular one of the identified playback devices, as annotated in the image to the right. Ex. 12; Ex. 1 (Bhattacharjee Decl.), ¶30.

Al-Shaykh (U.S. Publication No. 2011/0131520): And as yet a further example, patents such as Al-Shaykh disclosed a controller with a "send" button (image on the right) that could be used for "transfer of media content from the media application to [a] target rendering device in the home network." For example the user could select a "living room" (Fig. 3) or "bedroom" device (Fig. 4) for the transfer. Al-Shaykh at Abstract, ¶¶99-100 ("media transfer control 51" may be labeled "send," "transfer," "Play To," etc.).





In view of any one of the YTR Patent, Project Tungsten, Apple's Airplay or Al-Shayk patent publication, modifying the YTR prior art to display each of the paired devices that are available was nothing more than the application of a "known technique" (a user interface that displays available devices) to improve similar devices in the same way (improve YTR prior art by allowing selection of individual playback devices for transfer). Ex. 1, ¶165. A POSITA would have been motivated to make this straightforward and narrow modification, and would have had a reasonable expectation of success in doing so. Ex. 1 (Bhattacharjee Decl.), ¶170.

Accordingly, the YT prior art anticipates, or at minimum, renders obvious Claim 13.

C. Google Does Not Infringe Claim 1 of the '885 Patent

Asserted claim 1 of the '885 Patent requires that the accused system use what the patent calls a "zone scene," but the accused Google products cannot infringe because they do not include any

such "zone scene." Specifically, claim 1 requires a zone player (e.g., a speaker) to receive an indication that it has been added to a first "zone scene" and then to a second "zone scene." *See* '885 Pat. Cl. 1. Afterwards, the zone player will receive an "instruction to operate in accordance with a given one of the first and second zone scenes." *Id*. Sonos argues that the "zone scene" limitations are met because the accused Google speakers may be ***grouped*** together for playback. But a "zone scene" must be something more than merely a group. Rather, "zone scene" was a new term coined by the inventors that has been construed to mean "a previously saved grouping of zone players according to a common theme." Dkt. 106 (Markman Hearing Tr.) at 38:1-3. Because there is no genuine dispute that Google's products do not include a "common theme" or "zone scene," summary judgment of non-infringement is warranted.

What Sonos accuses as meeting the "zone scene" element are "groups" accessible through Google's Home application. But these are merely generic speaker groups, which were well known in speaker systems at the time. Ex. 14 (Schonfeld Decl.) ¶19-20. Using Google's Home application, a user may change the volume of the groups, and a user may play music through the grouped speakers or individual speakers, but Google never offers the user the option to add or change "scene" or "theme" information regarding any speaker or group.

The key claim term, "zone scene," has already been construed in this case as requiring a "common theme." Dkt. 106 at 38 ("The construction for that claim term is going to be: A previously-saved group of zone players according to a common theme."). As Google explained in its claim construction briefing, this construction should continue to bind Sonos because it is law of the case and Sonos has neither established that the construction was "clearly erroneous" nor presented any "new evidence" to warrant departing from the prior construction. *See* Dkt. 200 at 1-5. Consistent with the current construction, the specification clearly defines what "zone scene" means: "Using ***what is referred to herein as a theme or a zone scene***, zones can be configured in a particular scene (*e.g.*, morning, afternoon, or garden), where a predefined zone grouping and setting of attributes for the grouping are automatically effectuated." '885 Pat. at 8:47-51. Accordingly, a "scene" may automatically trigger a group with a particular playlist, volume, equalization, or other "attributes" described in the specification that are consistent with the common

theme. *See id.* at 9:20-30. Sonos's contrary construction that a "zone scene" may merely be a group of speakers reads out the critical meaning of the phrase—a "zone scene" must include some type of "scene" information. Omitting "scene" or "theme" information from the construction of "zone scene" would be contrary to the specification of the '885 patent, which repeatedly discusses "conventional multi-zone audio systems" and improvements to those audio systems using zone scenes. *Id.* at 1:46-2:24.

As discussed above, Google's accused products do not use "zone scenes" that must have a "common theme"—they just use conventional prior art speaker groups. Recognizing that there is no "theme" in the accused products, Sonos tries to manufacture two infringement theories to attempt to skirt the issue. First, Sonos's contentions argue that every generic speaker group has a "common theme" by speculating that "every 'speaker group' that a user creates in a Cast-enabled playback system has some common theme, which in this context amounts to ***whatever common topic, subject, etc. led the user to decide*** that these particular Cast-enabled media players should be placed into a previously-saved group that allows for synchronous playback when invoked." In other words, Sonos argues that the "common theme" is whatever the user was thinking about when he or she created the group. This argument is a non-starter because it is impossible to tell what a user was thinking to prove infringement and the case law decidedly rejects any argument that one can infringe by ***thinking***. For example, in *Amazon.com, Inc. v. Barnesandnoble.com, Inc.*, 239 F.3d 1343, 1353 (Fed. Cir. 2001), the Federal Circuit held that the plaintiff failed to establish a likelihood of success on infringement because its infringement theory required "assign[ing] a meaning to a patent claim that depends on the ***state of mind*** of the accused infringer." *Id.* (emphasis added); *see also Bd. of Trustees of Leland Stanford Junior Univ. v. Roche Molecular Sys., Inc.*, 528 F. Supp. 2d 967, 978 (N.D. Cal. 2007). Further, to the extent Sonos argues that ***every time*** a user creates a group, there must necessarily always be a "common theme" or "scene" associated with that group, doing so would effectively render the claim term "zone scene"—that Sonos coined—meaningless, which is a "highly disfavored" approach to claim construction. *Wasica Fin. GmbH v. Cont'l Auto. Sys., Inc.*, 853 F.3d 1272, 1288 (Fed. Cir. 2017).

Second, Sonos's contentions argue that because the accused Google Home app allows "the

user to input a name for the 'speaker group,'" this "serves as the user's shorthand label of the common theme." In other words, Sonos contends that anytime a group is named, it must meet the "common theme" requirement of the claims. This argument is similarly meritless. First of all, prior art systems, including those before the Examiner, allowed users to set the name of speaker groups. Ex. 14 (Schonfeld Decl.) ¶21. Even the admitted prior art in the patent describes zones named morning, evening, and weekend. '885 Pat. at 2:9-14; *PharmaStem Therapeutics, Inc. v. ViaCell, Inc.*, 491 F.3d 1342, 1362 (Fed. Cir. 2007) ("Admissions in the specification regarding the prior art are binding on the patentee . . . .").

Further, a user may name speaker groups in the Google Home app whatever he or she wants. For example, a user may name their three speaker groups "1," "2," and "3"; or "A," "B," and "C." None of these speaker group names indicates a "common theme," and none of these speaker groups is "serving as a user's shorthand label of the common theme" either. *See id*. Rather, users could name their speaker groups in the most expedient, but least descriptive way possible. They could also do the opposite, but in either case there is no reason to believe that a name of a speaker group is a "common theme."

Indeed, the specification confirms that "naming" groups is different from the claimed "zone scenes." "Zone scenes" are described as configuring zones "in a particular scene (e.g., morning, afternoon, or garden), where a predefined zone grouping ***and setting of attributes for the grouping are automatically effectuated***." '885 Pat. at 8:47-51 (emphasis added). On the other hand, the specification discusses naming speaker groups separately from the more complex "zone scene" settings; for example, describing a conventional set of a "morning group," "evening group," and "weekend group," each of which contained a speaker in the "den." '885 Pat. at 2:9-15.

In sum, "zone scene" is a term that the inventors coined to describe the allegedly novel aspects of the appropriately-titled "Zone Scene Management" patents. The inventors were aware that conventional speaker groups were in the prior art, and that those groups could be given names, as the background to the patent makes abundantly clear. Accordingly, the term "zone scene" was construed to require an additional "common theme" disclosed in the patent that could be applied to speaker groupings. That "common theme" is not met through the thoughts in the mind of a user as

he or she creates a conventional speaker group, nor is it merely the name of a speaker group. The former is unknowable and unprovable, and the latter is functionality that the patent distinguished from "zone scene" in the specification. Sonos has not identified any functionality aside from speaker groups and naming as meeting the "zone scenes" claim element, but neither are sufficient. As such, there is no genuine dispute of fact that Google's accused products do not have "zone scenes" and Google cannot infringe claim 1 of the '885 patent.

IV. CONCLUSION

For the foregoing reasons, Google respectfully requests that the Court grant Google's motion for summary judgment with respect to both claim 13 of the '615 Patent and claim 1 of the '885 Patent.

DATED: April 14, 2022

QUINN EMANUEL URQUHART & SULLIVAN, LLP

By: */s/ Charles K. Verhoeven*

Charles K. Verhoeven (Bar No. 170151)
charlesverhoeven@quinnemanuel.com
Melissa Baily (Bar No. 237649)
melissabaily@quinnemanuel.com
Lindsay Cooper (Bar No. 287125)
lindsaycooper@quinnemanuel.com
50 California Street, 22nd Floor
San Francisco, California 94111-4788
Telephone: (415) 875-6600
Facsimile: (415) 875-6700

Attorneys for GOOGLE LLC

CERTIFICATE OF SERVICE

Pursuant to the Federal Rules of Civil Procedure and Local Rule 5-1, I hereby certify that, on April 14, 2022, all counsel of record who have appeared in this case are being served with a copy of the foregoing via the Court's CM/ECF system and email.

/s/ *Charles K. Verhoeven*
Charles K. Verhoeven